NUMERICS USER GUIDE

R. VERFÜRTH

Contents

1. General remarks 1
2. Interpolation 3
3. Integration 4
4. Zeros of functions 5
5. Direct solvers for linear systems of equations 6
6. Iterative solvers for linear systems of equations 7
7. Eigenvalue problems 8
8. Initial value problems (fixed step-size) 10
9. Initial value problems (variable step-size) 11
10. Linear programs 13

1. General remarks

Numerics is an applet demonstrating some basic numerical algorithms. Upon starting the applet you are provided with a list of choices showing the implemented methods (see Fig. 1). You choose one and than click the **Run** button to start the chosen algorithm. Clicking the **Quit** button terminates the applet.

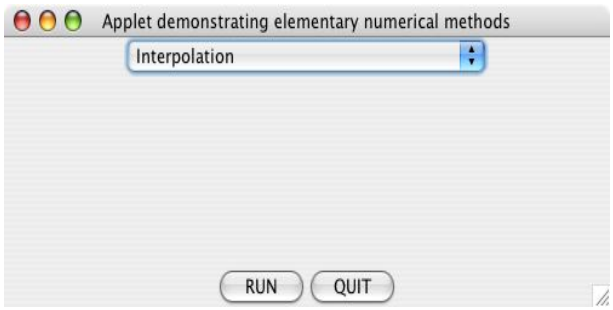

Figure 1. Numerics main window

Once you have started any of the implemented methods you'll see a screen (see Fig. 2) which is split into three parts: a left upper part labelled **Input** a right upper part labelled **Output** and a lower part containing buttons labelled **?**, **Compute**, **Draw Graph**, and **Quit**.

Date: September 2006.

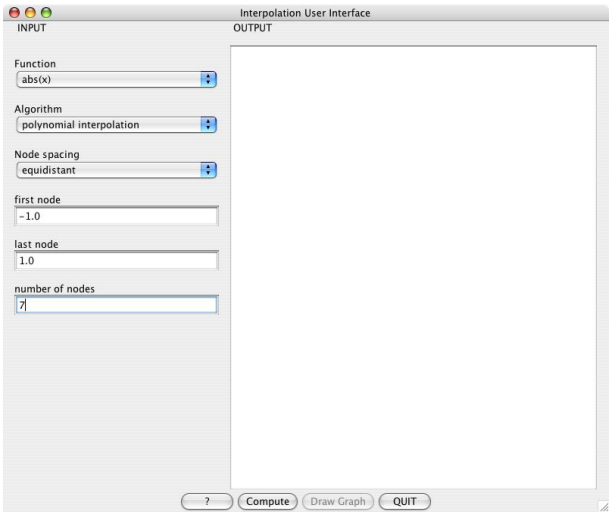

FIGURE 2. Window of a Numerics method

In the *input part* you'll see several choices that allow to choose examples, implemented methods etc. and text fields for setting relevant parameters. The text fields usually exhibit the default values of the corresponding parameters.

Once you have made your choices and entered your favourite parameters you press the **Compute** button to start the computations. The results will be depicted in the *output part*.

Note that some examples, e.g. interpolation of a rational function, require additional input. This will be provided via additional windows that show up **after** pressing the **Compute** Button of the main window. These additional windows all have a top label explaining their purpose, one or several labelled text fields for entering the relevant parameters, and an **OK** button. You have to press the latter one after having finished your input in order to continue the computation process.

Once the computation is completed you may eventually want to visualise the results by clicking the **Draw Graph** button. Methods that do not provide this option, e.g. Integration, have the **Draw Graph** button disabled. Upon pressing the **Draw Graph** button a graphics window will show up (see Fig. 3). It is split into three parts: an upper part containing text fields labelled **Xmin**, **Xmax**, **Ymin**, and **Ymax**, a lower part containing buttons labelled **Draw** and **Quit**, and a large middle part for the graphics. The values in the text fields determine the drawing area $[x_{\min}, x_{\max}] \times [y_{\min}, y_{\max}]$ and are initially set to default values depending on the actual method. *Note* that abscissa and ordinate scale separately, thus a circle will look like an ellipse if $x_{\max} - x_{\min}$ differs from $y_{\max} - y_{\min}$ Clicking the **Draw** button starts the graphics. Clicking the **Quit** button terminates the graphics, closes the graphics window and leads back to the main window of the actual method.

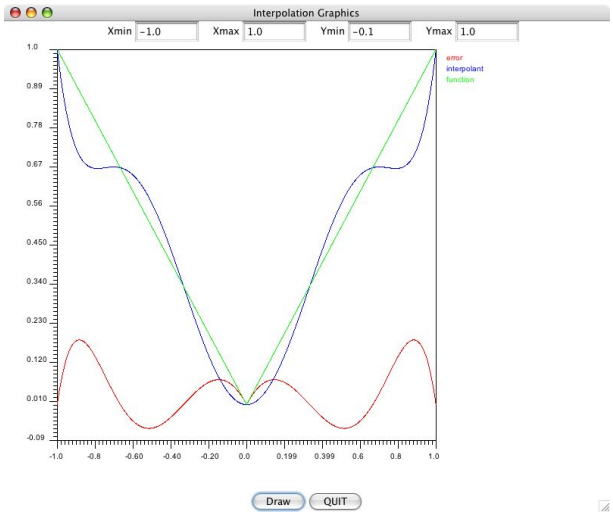

FIGURE 3. A sample graphics window

A new set of examples can be computed by adjusting the relevant choices and text fields and than clicking the **Compute** button.

Upon clicking the **?** button you'll find in the output area a short explanation of the actual method, of its properties, and of its functionality.

Clicking the **Quit** button terminates the actual method, closes the window, and leads back to the main window for eventually choosing a new numerical method.

2. INTERPOLATION

This method demonstrates various interpolation methods for functions of one real variable.

The choice **Function** provides the following sample functions for interpolation:

- $|x|$,
- $\sqrt{|x|}$,
- $\arctan(x)$,
- Pulse: 2π- periodic continuation of the characteristic function of $[0, \pi]$,
- Sawtooth: π-periodic continuation of the function $\pi - x$ on $[0, \pi]$,
- $\sin(Nx - a)$,
- Gaussian: normal distribution with user specified mean value μ and standard deviation σ,
- Polynomial: you will be asked for the degree and the coefficients,

- Rational function: you will be asked for the degrees and coefficients of the nominator and denominator polynomials.

The choice **Algorithm** provides the following interpolation methods:

- polynomial interpolation,
- spline interpolation (natural cubic splines),
- trigonometric interpolation.

The choice **Node spacing** provides the following options:

- equidistant: nodes are

$$x_{\text{first}} + \frac{i}{n-1}(x_{\text{last}} - x_{\text{first}}), \quad i = 0, 1, \dots, n-1,$$

- transformed Čebysev nodes: nodes are

$$x_{\text{last}} - \frac{\cos(\frac{i\pi}{n-1}) + 1}{2}(x_{\text{last}} - x_{\text{first}}), \quad i = 0, 1, \dots, n-1.$$

Note that when using trigonometric interpolation the Numerics applet automatically converts your input to an odd number of equidistant interpolation nodes.

In the text fields **First node** and **Last node** you should enter the first interpolation node x_{first} and the last interpolation node x_{last} respectively.

The text field **Number of nodes** is used to enter the number n of interpolation nodes.

Upon clicking the **Draw Graph** button you'll see a graphics window for visualising the error in red, the interpolation in blue, and the function to be interpolated in green.

3. Integration

This method demonstrates the numerical computation of integrals of functions of one real variable on a bounded interval with the use of quadrature formulae. With exception of the Romberg scheme all quadrature formulae are composite ones, i.e. the integration interval is split into a specified number of sub-intervals of equal length to which the given quadrature formula is applied.

The choice **Function** provides the following sample functions for integration:

- $\sqrt{|x|}$,
- $\arctan(x)$,
- $\sin(Nx - a)$,
- Gaussian: normal distribution with user specified mean value μ and standard deviation σ,
- Polynomial: you will be asked for the degree and the coefficients,
- Rational function: you will be asked for the degrees and coefficients of the nominator and denominator polynomials.

The choice **Algorithm** provides the following integration methods:

- Romberg scheme,
- Midpoint rule,
- Trapezoidal rule,
- Simpson rule,
- 2-point Gaussian rule,
- 3-point Gaussian rule,
- 4-point Gaussian rule.

In the text fields **left boundary** and **right boundary** you should enter the lower and upper integration bounds respectively.

The text field **initial number of subintervals** specifies the initial number of sub-intervals for the composite quadrature formula. The number of sub-intervals is doubled in each refinement step up to the maximal number of refinements specified in the text field **number of refinements**. When using the Romberg scheme the number given in this field also specifies the number of extrapolation columns.

4. Zeros of functions

This method demonstrates several algorithms for computing real zeros of functions of one real variable.

The choice **Function** provides the following sample functions:

- $\arctan(x)$,
- $\sin(Nx - a)$,
- Polynomial: you will be asked for the degree and the coefficients,
- Rational function: you will be asked for the degrees and coefficients of the nominator and denominator polynomials.

The choice **Algorithm** provides the following methods:

- Newton method without damping,
- Newton method with damping,
- Secant rule,
- Regula falsi.

In the text field **initial guess for zero** you must enter the starting value for the corresponding algorithm. The secant rule and the regula falsi both require a second starting value. This one must be provided in the text field **second guess for zero (secant rule and regula falsi)**. Please ignore this field when using a Newton method. Recall that the function values at the two initial guesses must be of opposite sign when using the regula falsi.

In the text field **number of iterations** you should provide an upper bound for the number of iterations. In the text field **tolerance** you are asked to specify an error tolerance. Iterations will stop when either the given maximal number of iterations is achieved or when the absolute

value of the function value at the actual iterate is below the prescribed tolerance.

5. Direct solvers for linear systems of equations

This method provides algorithms for solving linear systems of equations using a direct solver, i.e. Gaussian elimination and its relatives.

The choice **Initialization of LSE** gives you the following options for entering the linear system of equations to be solved:

- Manual: You will be asked for the coefficients of the matrix and the entries of the right-hand side. These are entered via additional windows that will show up **after** pressing the **Compute** button of the actual window.
- 5-point stencil with random rhs: Here the matrix is the one corresponding to a regular 5-point difference approximation of the Laplacian on a uniform grid, i.e.

  $$4u_{\text{center}} - u_{\text{north}} - u_{\text{east}} - u_{\text{south}} - u_{\text{west}} = h^2 f_{\text{center}}.$$

  The number of gridlines is the largest integer not greater than the square root of the prescribed size of the linear system. The right-hand side of the linear system is generated with a random-number generator.
- Random spd matrix, random rhs: Here matrix and right-hand side both are generated with a random-number generator. The diagonal entries of the matrix have the form $(0.5 + \text{rand}) \cdot \text{dim}$; the lower-diagonal entries take the form $\text{rand} - 0.5$; the upper-diagonal entries equal their lower-diagonal counterparts. Here, dim is the number of equations and rand a random-number generator with values in the interval $[0, 1]$. This construction yields a symmetric and positive definite matrix.

The choice **Algorithm** allows to choose among of the following numerical methods:

- Gaussian elimination,
- LR-decomposition (also called LU-decomposition),
- Cholesky decomposition (only applicable to symmetric positive definite matrices),
- QR-decomposition.

The choice **Output** provides the following options:

- Norm of final residual: Euclidean norm of the residual of the computed solution.
- Solution vector: Components of the computed solution.
- Matrix, rhs, solution: Matrix entries and components of the right-hand side and of the computed solution.

In the text field **number of columns/rows** you should enter the number of equations of the linear system.

6. Iterative solvers for linear systems of equations

This method provides algorithms for approximately solving linear systems of equations using a stationary iterative method.

The choice **Initialization of LSE** gives you the following options for entering the linear system of equations to be solved:

- Manual: You will be asked for the coefficients of the matrix and the entries of the right-hand side. These are entered via additional windows that will show up **after** pressing the **Compute** button of the actual window.
- 5-point stencil with random rhs: Here the matrix is the one corresponding to a regular 5-point difference approximation of the Laplacian on a uniform grid, i.e.

  $$4u_{\text{center}} - u_{\text{north}} - u_{\text{east}} - u_{\text{south}} - u_{\text{west}} = h^2 f_{\text{center}}.$$

  The number of gridlines is the largest integer not greater than the square root of the prescribed size of the linear system. The right-hand side of the linear system is generated with a random-number generator.
- Random spd matrix, random rhs: Here matrix and right-hand side both are generated with a random-number generator. The diagonal entries of the matrix have the form $(0.5 + \text{rand}) \cdot \text{dim}$; the lower-diagonal entries take the form $\text{rand} - 0.5$; the upper-diagonal entries equal their lower-diagonal counterparts. Here, dim is the number of equations and rand a random-number generator with values in the interval $[0, 1]$. This construction yields a symmetric and positive definite matrix.

The choice **Algorithm** allows to choose among of the following numerical methods:

- Richardson iteration,
- Jacobi iteration,
- Gauss-Seidel iteration,
- Symmetric successive over-relaxation (SSOR),
- Gradient iteration,
- Conjugate gradient iteration (CG),
- Conjugate gradient iteration with SSOR-preconditioning (SSOR PCG).

The choice **Output** provides the following options:

- Norm of final residual: Euclidean norm of the residual of the computed solution.
- Norm of all residuals: Euclidean norm of the residuals of all iterates.
- Solution vector: Components of the computed solution.
- Matrix, rhs, solution: Matrix entries and components of the right-hand side and of the computed solution.

In the text field **number of columns/rows** you should enter the number of equations of the linear system.

In the text fields **number of iterations** and **tolerance** you are asked to prescribe the maximal number of iterations and the desired error tolerance. Iterations will stop either when the maximal number of iterations has been performed or when the Euclidean norm of the residual of the actual iterate is below the prescribed tolerance.

Once the computations have successfully been performed you may visualise the results by pressing the **Draw Graph** button. This will open a new graphics window. It depicts the following three curves:

- Red: relative residual, i.e. the ratio of the actual residual to the initial residual.
- Blue: convergence rate, i.e. the ratio of the actual residual to the previous one.
- Green: mean convergence rate, i.e. the i-th root of the ratio of the actual residual to the initial residual, where i denotes the actual number of iterations.

7. Eigenvalue problems

This method provides algorithms for computing the largest eigenvalue, the smallest eigenvalue, and all eigenvalues of a square matrix.

The choice **Initialization of matrix** gives you the following options for entering the linear system of equations to be solved:

- Manual: You will be asked for the coefficients of the matrix. These are entered via additional windows that will show up **after** pressing the **Compute** button of the actual window.
- 5-point stencil: Here the matrix is the one corresponding to a regular 5-point difference approximation of the Laplacian on a uniform grid, i.e.

  $$4u_{\text{center}} - u_{\text{north}} - u_{\text{east}} - u_{\text{south}} - u_{\text{west}} = h^2 f_{\text{center}}.$$

  The number of gridlines is the largest integer not greater than the square root of the prescribed size of the linear system.
- Random spd matrix: Here matrixis generated with a random-number generator. The diagonal entries of the matrix have the form $(0.5 + \text{rand}) \cdot \text{dim}$; the lower-diagonal entries take the form $\text{rand} - 0.5$; the upper-diagonal entries equal their lower-diagonal counterparts. Here, dim is the number of equations and rand a random-number generator with values in the interval $[0, 1]$. This construction yields a symmetric and positive definite matrix.

The choice **Algorithm** allows to choose among of the following numerical methods:

- Power iteration,

- Rayleigh quotient iteration (only for symmetric positive definite matrices),
- Inverse power iteration,
- Inverse Rayleigh quotient iteration (only for symmetric positive definite matrices),
- QR iteration.

The first two methods yield approximations of the largest (in absolute value) eigenvalue, the third and fourth ones those of the smallest (in absolute value) eigenvalue. The last method gives approximations of all eigenvalues. In this method the matrix is first transformed to a similar one with upper Hessenberg form. This one will be a tridiagonal matrix if the original matrix is symmetric.

The choice **Output** offers the following options:

- Final approximation of eigenvalue(s): Computed eigenvalue(s) after the last iteration.
- All approximations of eigenvalue(s): Computed eigenvalue(s) after every iteration.
- Matrix and final approximation of eigenvalue(s): Matrix and computed eigenvalue(s) after the last iteration.
- Matrix and all approximations of eigenvalue(s): Matrix and computed eigenvalue(s) after every iteration.

In the text field **number of columns/rows** you should enter the size of the matrix.

In the text fields **number of iterations** and **tolerance** you are asked to prescribe the maximal number of iterations and the desired error tolerance. Iterations will stop either when the maximal number of iterations has been performed or when the difference between the actual approximation and the previous one is in absolute value below the prescribed tolerance. When using the QR algorithm "approximation"; means the largest absolute value in the lower diagonal of the current matrix.

Once the computations have successfully been performed you may visualise the results by pressing the **Draw Graph** button. This will open a new graphics window. It depicts the following three curves:

- Red: relative residual, i.e. the ratio of the estimated actual error to the initial estimated error.
- Blue: convergence rate, i.e. the ratio of the actual estimated error to the previous one.
- Green: mean convergence rate, i.e. the i-th root of the ratio of the actual estimated error to the initial estimated error, where i denotes the actual number of iterations.

Here the error is always estimated by computing a three-term extrapolation of the computed approximation to the corresponding eigenvalue.

When using the QR algorithm "approximation" means the largest absolute value in the lower diagonal of the current matrix.

8. Initial value problems (fixed step-size)

This method provides algorithms for numerically solving an initial value problem. All algorithms use a fixed step-size.

At the top of the Input section you'll find a list of labelled checkboxes showing the implemented algorithms:

- Explicit Euler scheme,
- Implicit Euler scheme,
- Crank-Nicolson scheme (also called trapezoidal rule),
- Classical Runge-Kutta scheme (explicit, order 4),
- Strongly diagonal implicit Runge-Kutta scheme of order 3 (SDIRK3, 2 levels),
- Strongly diagonal implicit Runge-Kutta scheme of order 4 (SDIRK4, 5 levels),
- Strongly diagonal implicit Runge-Kutta scheme of order 5 (SDIRK5, 6 levels),
- Backward difference scheme with 2 steps (BDF2),
- Backward difference scheme with 3 steps (BDF3),
- Backward difference scheme with 4 steps (BDF4).

You choose an algorithm by clicking its checkbox. You may choose as many algorithms as you like, but at least one. Note that the implicit Euler, the Crank-Nicolson and the SDIRK schemes are A-stable and that the BDF schemes are A_0-stable.

The choice **ODE** provides the following sample differential equations:

- Population model:

$$x' = ax - bxy$$
$$y' = cy - dxy$$

  with positive parameters a, b, c, d that must be provided via an additional window that shows up **after** pressing the **Compute** button of the actual window. The differential equation has a stationary point at $x = \frac{c}{d}$, $y = \frac{a}{b}$ which is attained if the initial value equals this value. Otherwise the solution curve is a closed periodic orbit.
- Linear system: $x' = Ax$, you will be asked for the dimension of the matrix A and for its entries.
- $y' = ay$ (a real): You will be asked for a.
- $y' = ay$ (a complex): This is discretized as the real system

$$u' = \mathrm{Re}(a)u - \mathrm{Im}(a)v$$
$$v' = \mathrm{Im}(a)u + \mathrm{Re}(a)v$$

with $u = \mathrm{Re}(y)$ and $v = \mathrm{Im}(y)$. You will be asked for the real part $\mathrm{Re}(a)$ and the imaginary part $\mathrm{Im}(a)$ of the complex number a.

- $y'' + dy' + \omega y = a\sin(\varphi t - \psi)$: This is discretized as the corresponding first order system

$$x' = a\sin(\varphi t - \psi) - dx - \omega y$$
$$y' = x.$$

You will be asked for the numbers d, ω, a, φ, and ψ.

With the choice **Output** you determine the amount of printed output in the Output area. You have the options:

- None: No output.
- All approximations: Computed approximations after every time step.

In the text fields **initial time** and **final time** you should specify the initial time t_0 and the final time T.

In the text field **number of steps** you are asked to give the desired number n of steps to be performed. The fixed step-size then is $\frac{T-t_0}{n}$.

Once the computations have successfully been performed you may visualise the results be pressing the **Draw Graph** button. This will open a new graphics window depicting the computed solution curves for all chosen algorithms.

9. Initial value problems (variable step-size)

This method provides algorithms for numerically solving an initial value problem using step-size control. The Runge-Kutta-Fehlberg methods use a step-size control based on comparing schemes with different order. The other methods use a step-size control based on comparing results using a given method with step-sizes h and $\frac{h}{2}$.

At the top of the Input section you'll find a list of labelled checkboxes showing the implemented algorithms:

- Explicit Euler scheme,
- Implicit Euler scheme,
- Crank-Nicolson scheme (also called trapezoidal rule),
- Classical Runge-Kutta scheme (explicit, order 4),
- Strongly diagonal implicit Runge-Kutta scheme of order 3 (SDIRK3, 2 levels),
- Strongly diagonal implicit Runge-Kutta scheme of order 4 (SDIRK4, 5 levels),
- Strongly diagonal implicit Runge-Kutta scheme of order 5 (SDIRK5, 6 levels),
- Runge-Kutta-Fehlberg scheme of order 2 with 4 levels (RKF2),
- Runge-Kutta-Fehlberg scheme of order 3 with 5 levels (RKF3),
- Runge-Kutta-Fehlberg scheme of order 4 with 7 levels (RKF4).

You choose an algorithm by clicking its checkbox. You may choose as many algorithms as you like, but at least one. Note that the implicit Euler, the Crank-Nicolson and the SDIRK schemes are A-stable. The Runge-Kutta-Fehlberg methods are all explicit schemes and have bounded stability regions.

The choice **ODE** provides the following sample differential equations:

- Population model:

$$x' = ax - bxy$$
$$y' = cy - dxy$$

  with positive parameters a, b, c, d that must be provided via an additional window that shows up **after** pressing the **Compute** button of the actual window. The differential equation has a stationary point at $x = \frac{c}{d}$, $y = \frac{a}{b}$ which is attained if the initial value equals this value. Otherwise the solution curve is a closed periodic orbit.
- Linear system: $x' = Ax$, you will be asked for the dimension of the matrix A and for its entries.
- $y' = ay$ (a real): You will be asked for a.
- $y' = ay$ (a complex): This is discretized as the real system

$$u' = \mathrm{Re}(a)u - \mathrm{Im}(a)v$$
$$v' = \mathrm{Im}(a)u + \mathrm{Re}(a)v$$

  with $u = \mathrm{Re}(y)$ and $v = \mathrm{Im}(y)$. You will be asked for the real part $\mathrm{Re}(a)$ and the imaginary part $\mathrm{Im}(a)$ of the complex number a.
- $y'' + dy' + \omega y = a\sin(\varphi t - \psi)$: This is discretized as the corresponding first order system

$$x' = a\sin(\varphi t - \psi) - dx - \omega y$$
$$y' = x.$$

  You will be asked for the numbers d, ω, a, φ, and ψ.

With the choice **Output** you determine the amount of printed output in the Output area. You have the options:

- None: No output.
- All approximations: Computed approximations after every time step.

In the text fields **initial time** and **final time** you should specify the initial time t_0 and the final time T.

In the text field **maximal number of steps** you are asked to give the desired maximal number n of steps to be performed. The initial guess for the first step-size is $0.01\frac{T-t_0}{n}$.

In the text field **error tolerance** you have to prescribe the tolerance TOL for the step-size control. The step-size control tries to monitor the computation such that all approximations have an error of at most TOL when compared to the (unknown) solution of the differential equation.

Once the computations have successfully been performed you may visualise the results be pressing the **Draw Graph** button. This will open a new graphics window depicting the computed solution curves for all chosen algorithms.

10. Linear programs

This method provides algorithms for solving linear programs in standard form: minimize $c^T x$ subject to the constraint $Ax = b$, $x \geq 0$.

The choice **Initialization of LP** gives you the following options for entering the linear system of equations to be solved:

- Manual: You will be asked for the coefficients of the matrix A and the entries of the vectors b and c. These are entered via additional windows that will show up **after** pressing the **Compute** button of the actual window. The number of columns of A has to fit with the value in the text field **number of parameters** and the number of rows of A has to be the same as the value in the text field **number of constraints**.
- Example shoes: This example corresponds to the linear program in non-standard form: maximize $16x + 32y$ subject to the constraints

$$\begin{aligned} 20x + 10y &\leq 8000 \\ 4x + 5y &\leq 2000 \\ 6x + 15y &\leq 4500. \end{aligned}$$

- Example nutrition: This example corresponds to the linear program in non-standard form: minimize $10x + 7y$ subject to the constraints

$$\begin{aligned} 20x + 20y &\geq 60 \\ 15x + 3y &\geq 15 \\ 5x + 10y &\geq 20. \end{aligned}$$

- Example dual problem: This example corresponds to the linear program in non-standard form: minimize $x + y$ subject to the constraints

$$\begin{aligned} 2x + y &\geq 3 \\ x + 2y &\geq 3. \end{aligned}$$

- Example Klee-Minty: This example corresponds to the linear program in non-standard form: maximize x_n subject to the constraints $-a_n x_{i-1} \leq x_i \leq 1 - a_n x_{i-1}$ with $x_0 = 1$ and $a_n =$

$0.5^{(n+1)^2}$. You'll have to enter the number n in the text field **number of parameters**.

The choice **Algorithm** allows to choose among of the following numerical methods:

- Primal simplex: Apply the standard primal simplex algorithm starting from an admissible inital bases for the primal problem.
- Dual simplex: Apply the standard dual simplex algorithm starting from an admissible inital bases for the dual problem.
- Auto start simplex: In a first step the primal simplex method is applied to a suitable auxiliary linear program in order to obtain a first admissible bases for the original linear program. Then the primal simplex algorithm is applied to the original linear program starting with this initial bases.
- Inner point: A Newton iteration is applied to an auxiliary nonlinear optimization method which has the same solution as the original linear program.
- Nelder-Mead: This implements the Nelder-Mead algorithm of bisecting hyperplanes.

The choice **Initialization of Algorithm** determines the way in which the chosen algorithm is started. Some algorithms require an initial admissible bases. This may be entered either manually via an additional window or may be determined automatically. In the latter case the algorithms always take the last nr indices where nr is the number of rows of A. This device is well suited for linear programs that originally have the non-standard forms $Ax \leq b$ (primal simplex) or $Ax \geq b$ (dual simplex).

The choice **Output** provides the following options:

- Optimal value of $c^T x$.
- All values of $c^T x$.
- All values of of $c^T x$ and the solution vector x.
- All values of of $c^T x$, the solution vector x and the data of the linear program.
- All values of of $c^T x$, all iterates x and the data of the linear program.

In the text field **number of parameters** you should enter the number of unknowns.

In the text field **number of constraints** you should enter the number of constraints.

In the text fields **number of iterations** and **tolerance** you are asked to prescribe the maximal number of iterations and the desired error tolerance. Iterations will stop when the maximal number of iterations has been performed. In addition the inner-point iteration and

the Nelder-Mead algorithm will terminate when the Euclidean norm of a suitable residual is below the prescribed tolerance.

Ruhr-Universität Bochum, Fakultät für Mathematik, D-44780 Bochum, Germany

E-mail address: rv@num1.rub.de